Un modelo de interacción basado en conectores

Silvia Amaro
samaro@uncoma.edu.ar
Dpto. de Ciencias de la Computación, Universidad Nacional del Comahue, Argentina
Ernesto Pimentel
ernesto@lcc.uma.es
Dpto. de Lenguajes y Ciencias de la Computación, Universidad de Málaga,España
Ana M. Roldán
amroldan@diesia.uhu.es
Dpto. de Ingeniería Electrónica y Sistemas Informáticos. Universidad de Huelva, España

Resumen

Utilizar componentes significa comprender como interactúan de manera individual con su entorno y especificar como deberían ser sus interacciones mutuas y cooperativas para que su composición sea correcta [10]. Por otro lado, al trabajar con componentes heterogéneas, uno de los objetivos a lograr es conseguir una separación clara entre los aspectos de interacción y computación, para así favorecer la reutilización y el análisis global de la aplicación. En este contexto se manifiesta la necesidad de disponer de modelos de coordinación [3, 9] que puedan ser utilizados para describir explícitamente protocolos de coordinación complejos en términos de primitivas simples y constructores.
Reo es un modelo de coordinación basado en canales en el cual la comunicación sólo es posible en presencia de conectores [4]. Podemos considerar a Reo como un lenguage de adaptación utilizado para la construcción de conectores que instrumentan instancias de componentes en un sistema basado en componentes. Nuestra línea de investigación se centra en el estudio de la expresividad de Reo para utilizarlo como mecanismo para la descripción del comportamiento interactivo de componentes software. Proponemos enriquecer la información provista por los lenguajes de descripción de Interfaces (IDL's), con especificaciones basadas en Reo del comportamiento de las componentes, orientado a resolver problemas de interoperabilidad a nivel de protocolos.
Palabras clave: Componentes, coordinación, expresividad, interoperabilidad.

1. Introducción

En el desarrollo de nuevas aplicaciones a partir del ensamblado de componentes existentes, heterogéneas, posiblemente desarrolladas en distintos momentos, con distintos formatos, diseños e implementaciones, y provenientes de las más diversas fuentes, se requiere el uso de mecanismos de composición que permitan que la integración pueda llevarse a cabo de forma correcta. Al momento de seleccionar las componentes que intervendrán en la aplicación, deben considerarse

sus capacidades y las incompatibilidades que puedan surgir en su interacción. Es aquí donde la interoperabilidad juega un rol fundamental.

La interoperabilidad se refiere a la habilidad de dos o más entidades de comunicarse y cooperar a pesar de las diferencias en el lenguaje de implementación, el ambiente de ejecución, y la abstracción del modelo. Pueden distinguirse tres niveles de interoperabilidad:

1. a nivel *signatura*, se basa en la signatura de las operaciones que ofrecen las componentes (nombres, tipos de parámetros y valores de retorno).

2. a nivel *semántico*, la interoperabilidad se basa en el significado de las operaciones. se trata de que los proveedores y consumidores tengan un entendimiento común del significado de los servicios y datos requeridos.

3. a nivel *protocolo*, se refiere al orden relativo en que una entidad espera que sus servicios sean solicitados, el orden en que ella solicita servicios a otras entidades, y las condiciones de bloqueo y reglas que gobiernan las interacciones entre entidades.

Por otro lado se requiere que la composición sea altamente flexible y que el todo, es decir la aplicación resultante, sea mas que solo la suma de sus partes. En este escenario la responsabilidad parece recaer sobre el código adaptador (glue code) cuya finalidad es conectar componentes, facilitar su comunicación, y coordinar sus interacciones. Se pretende que el código adaptador contribuya a la semántica de la aplicación, conservando su independencia del dominio de la aplicación, y así, dos composiciones del mismo conjunto de componentes podrían resultar en sistemas diferentes. Además debe existir una separación clara entre la especificación de las componentes computacionales y la especificación de sus interacciones. Esta separación facilita la reutilización de las componentes y también la reutilización de los patrones de comunicación.

Los modelos y lenguajes de coordinación ofrecen diversas soluciones al problema de especificar e instrumentar las interacciones entre instancias de componentes independientes y distribuidas. En el caso de los modelos de coordinación basados en canales, el framework evoluciona por medio de la aplicación de acciones de comunicación sobre extremos de entrada o salida de canales a los que están conectadas las componentes que están siendo coordinadas. Reo [...] es un modelo de coordinación basado en canales que sugiere el uso de conectores para la coordinación de processos concurrentes o instancias de componentes en un sistema basado en componentes. Cuando Reo fue introducido, Arbab [...] mostró la potencia expresiva de su propuesta por medio de varios ejemplos, simulando en una forma simple y elegante diferentes mecanismos de comunicación.

Nuestra propuesta es utilizar Reo como mecanismo para la descripción del comportamiento interactivo de componentes software. Proponemos complementar la información provista por los lenguajes de descripción de Interfaces (IDL's) ofrecidos por las plataformas orientadas a componentes del mercado, con especificaciones basadas en Reo del comportamiento de las componentes, orientado a resolver problemas de interoperabilidad a nivel de protocolos. A continuación introducimos brevemente el modelo y damos algunas conclusiones.

2. Reo, un modelo de coordinación basado en canales

.

Reo es un modelo de coordinación que permite la construcción de sistemas por la composición de componentes que interaccionan a través de conectores.Los conectores se construyen a partir de un conjunto de canales, síncronos y asíncronos con comportamiento bien definido. Si bien Reo sugiere un conjunto básico de canales, no impone restricciones respecto a los tipos y comportamiento de los canales, lo que permite la definición por parte del usuario de nuevos tipos de canales con comportamiento específico. El mecanismo de composición de canales junto a la disponibilidad de una amplia variedad de tipos de canales con semánticas diferentes de las tradicionales, permiten la construcción de un gran número de conectores que imponen patrones de coordinación específicos muy interesantes. En este contexto Reo se manifiesta como un modelo muy flexible y potente. Un conector puede utilizarse como el código adaptador necesario en la composición de componentes, tiene su propia semántica, y ante conectores diferentes pueden obtenerse composiciones diferentes para un mismo conjunto de componentes. La comunicación entre instancias de componentes se logra exclusivamente por la aplicación de operaciones de entrada y salida sobre los puntos de conexión del conector a los que están conectadas. En [...] Arbab et al. introdujeron un modelo operacional para el comportamiento de los conectores de Reo basado en Automata de Restricciones (Constraint Automata). El automata observa las ocurrencias de datos sobre los extremos del conector y cambia su estado de acuerdo al dato observado, o lo rechaza si no existe en el automata una restricción que se corresponda.

3. Especificación de protocolos de interacción

3.1. El lenguage $\mathcal{R}$

Proponemos un álgebra de procesos $\mathcal{R}$ basado en las primitivas de comunicación de Reo . Los agentes en $\mathcal{R}$ se construyen a partir de los operadores prefijo, elección no determinista y composición paralela. A continuación definimos formalmente la sintaxis de $\mathcal{R}$:

$$\begin{aligned} P &::= 0 \mid A.P \mid P + P \mid P \parallel P \mid recX.P \\ A &::= wr(c,v) \mid tk(c,v) \mid rd(c,v) \end{aligned}$$

dónde 0 denota el proceso vacío, y c un extremo de entrada o salida de un conector. Los prefijos wr, tk y rd son las abreviaciones de las operaciones *write*, *take* y *read* respectivamente.
En la definición de la semántica operacional del cálculo, consideramos el comportamiento de un conector C dado por un conjunto de transiciones que tienen la forma

$$\langle \mathrm{C}, \overline{act} \rangle \overset{\overline{act}_1}{\longmapsto}_{\mathbb{C}} \langle \mathrm{C}', \overline{act}_2 \rangle$$

dónde $\overline{act}$ denota el conjunto de acciones que aplicadas en paralelo sobre los extremos del conector pueden producir un progreso sobre el mismo, dando un nuevo estado representado por C′. El conjunto $\overline{act}_1$ denota las acciones cuya aplicación tiene éxito inmediato, y $\overline{act}_2$ representa el subconjunto de acciones que han quedado suspendidas en algún extremo del conector. Un ejemplo de un conector simple en Reo es $LR = \{c_i, c_o\}$, las transiciones (1), (2) y (3) describen su comportamiento.

$$\langle \{c_i, c_o\}, \{wr(c_i, d)\} \rangle \overset{\{wr(c_i,d)\}}{\longmapsto}_{\mathbb{C}} \langle \{{c_i}', c_o\}, \emptyset \rangle \tag{1}$$

$$\langle \{{c_i}^k, {c_o}^j\}, \{tk(c_o, d)\} \rangle \overset{\{tk(c_o,d)\}}{\longmapsto}_{\mathbb{C}} \langle \{{c_i}^k, {c_o}^{j+1}\}, \emptyset \rangle \quad (k > j) \tag{2}$$

$$\langle \{{c_i}^k, {c_o}^j\}, \{rd(c_o, d)\} \rangle \overset{\{rd(c_o,d)\}}{\longmapsto}_{\mathbb{C}} \langle \{{c_i}^k, {c_o}^j\}, \emptyset \rangle \quad (k > j) \tag{3}$$

4. Expresividad del modelo

Para analizar la potencia expresiva del lenguaje $\mathcal{R}$, en primera instancia lo comparamos con el lenguaje $\mathcal{L}$ propuesto en [1] para el modelo de coordinación Linda. El criterio adoptado para la comparación de ambos modelos se basa en la noción de inmersión modular propuesta por De Boer y Palamidessi en [6], o sea una traducción composicional que preserva algunos aspectos de observables, en particular éxito y fracaso.

Comparamos la potencia expresiva de $\mathcal{L}$ y $\mathcal{R}$ en presencia de un conector LR como el introducido en la sección previa. A continuación presentamos las nociones de observables y el compilador utilizado

5. Conclusions

Los trabajos realizados sobre lenguajes y modelos de coordinación tienen como objetivo principal la interoperabilidad de aplicaciones software, normalmente descritas en lenguajes de programación distintos. En este sentido el modelo de interacción presentado permite ampliar la información que proporcionan las interfaces de componentes con el fin de evitar multitud de problemas, como los que surgen a nivel de protocolos, y así asegurar que el comportamiento cooperativo será el esperado. Para esto último definiremos una noción de compatibilidad entre componentes especificadas segun el modelo.

Como consecuencia del estudio de la expresividad del modelo, observamos que, para el caso concreto presentado, ambos modelos son igualmente expresivos. Aunque el resultado principal obtenido era previsible, dado que el conector considerado en Reo permite simular las acciones de Linda sobre el espacio de tuplas, este trabajo constituye un punto de partida para analizar la expresividad entre las dos alternativas más influyentes en el contexto de los modelos de coordinación. El enfoque utilizado y la homogeneizacón de las dos familias de lenguages bajo cálculos similares permitirá realizar una comparación exhaustiva de ambos modelos de coordinación, analizando la expresividad de otros conectores de Reo frente a primitivas adicionales de Linda.

Nuestro trabajo futuro estará destinado a estudiar profundamente mas conectores de Reo y definir un lenguaje de descripción de interacciones basado en Reo para la coordinación de componentes al estilo de lo realizado en el contexto de Linda. A partir del análisis exahustivo de la expresividad de los dos modelos descritos, nuestra intención es explorar si uno es más adecuado que el otro en determinadas circunstancias para expresar la interacción entre componentes. No descartamos la posibilidad de que partes de un sistema puedan ser convenientemente descritos con un formalismo, mientras que otras sean más susceptibles de ser especificadas con el otro modelo.

Referencias

[1] S. Amaro, E. Pimentel and A.M. Roldán. Coordinación basada en tuplas compartidas y en canales de comunicación. In *Proceedings of PROLE'03*, Noviembre 2003.

[2] S. Amaro, E. Pimentel and A.M. Roldán. A Preliminary Comparative Study on the Expressive Power of Reo and Linda. In *FOCLASA 2004*, ENTCS (Electronics Notes in Theoretical Computer Science), pages ????, Agosto 2004

[3] F. Arbab. The IWIM model for coordination of concurrent activities. In *First International Conference on Coordination Models, Languajes and Applications (Coordination'96)*, LNCS, pages 34–56, 1996.

[4] F. Arbab. A Channel-based Coordination Model for Component Composition. In *Proceedings of the 16th European Conference on Object-Oriented Programming* 2002.

[5] F. Arbab and J.J.M.M. Rutten. A coinductive calculus of component connectors. *CWI Report SEN- R0216 ISSN 1386-369X* 2002.

[6] F.S. de Boer and C. Palamidesi. Embedding as a tool for language comparision. *In Information and Compt*, 108(1):128-157, 1991.

[7] A. Bracciali, A. Brogi, and F. Turini. Coordinating Interaction Patterns. In *Proceedings of 16th ACM Symposium on Applied Computing*, 2001.

[8] C. Canal, L. Fuentes, E. Pimentel, J. Troya, and A. Vallecillo. Extending Corba Interfaces with Protocols. *The Computer Journal*, 44(5):448–462, 2001.

[9] D. Gelernter and N. Carriero. Coordination Languages and Their Significance. *Communications of de ACM*, 35(3):97–107, 1992.

[10] G. T. Leavens and M. Staraman, editors. *Foundations of Component-Based Systems*. Cambridge University Press, 2000.

[11] A. Roldán, E. Pimentel and A. Brogi. Safe Composition of Linda-based Components, TACoS'03(ETAPS'03). Electronic Notes in Theoretical Computer Science, 82 No. 6, 2003.